Learning optical flow from still images

Filippo Aleotti* Matteo Poggi* Stefano Mattoccia
Department of Computer Science and Engineering (DISI)
University of Bologna, Italy
{filippo.aleotti2, m.poggi, stefano.mattoccia }@unibo.it

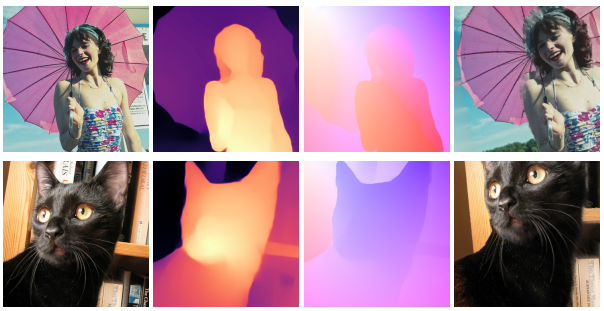
Abstract

This paper deals with the scarcity of data for training optical flow networks, highlighting the limitations of existing sources such as labeled synthetic datasets or unlabeled real videos. Specifically, we introduce a framework to generate accurate ground-truth optical flow annotations quickly and in large amounts from any readily available single real picture. Given an image, we use an off-the-shelf monocular depth estimation network to build a plausible point cloud for the observed scene. Then, we virtually move the camera in the reconstructed environment with known motion vectors and rotation angles, allowing us to synthesize both a novel view and the corresponding optical flow field connecting each pixel in the input image to the one in the new frame. When trained with our data, state-of-the-art optical flow networks achieve superior generalization to unseen real data compared to the same models trained either on annotated synthetic datasets or unlabeled videos, and better specialization if combined with synthetic images.

1. Introduction

The problem of estimating per-pixel motion between video frames, also known as *optical flow* [50], has a long history in computer vision and remains far from being solved. On top of it, several higher-level tasks such as tracking, action recognition and more are typically performed. Among the main challenges for optical flow systems, there are occlusions, motion blur and lack of texture.

Deep learning has played a crucial role in the latest years of research on this topic, at first to learn a data term [1, 65] and then to directly infer the dense optical flow field in end-to-end manner [7, 20, 51, 52, 18, 19, 17, 53], currently representing the state-of-the-art in this field. This achievement has been made possible by the availability of extensive training data labeled with ground-truth optical flow fields, most of them obtained through computer graphics



a) b) c) d)

Figure 1. ***Depthstillation* from still images.** From left to right: a) single input image, b) estimated depth map, c) optical flow field consequence of virtual camera motion, d) virtual view. We show b) as inverse depth to improve visualization.

[5, 7, 20]. Unfortunately, these large datasets alone are not enough to train a neural network for its deployment in real environments, because of the well-known *domain shift* occurring when moving from synthetic images to real ones. A notable example is represented by the KITTI optical flow benchmarks [10, 35], over which deep networks that have been trained only on synthetic data perform poorly, as witnessed by recent works [20, 51, 53]. This problem is known in literature and has been faced for other tasks such as semantic segmentation [15, 39, 43, 55] or stereo depth estimation [56, 57, 68, 61]. To fully restore a level of accuracy comparable to the one achieved on synthetic data, fine-tuning on imagery similar to the testing domain is usually required. Anyway, obtaining ground-truth optical flow labels for real images is particularly challenging because there exists virtually no sensor capable of acquiring ground-truth correspondences between points in challenging real-world scenes [37]. A viable strategy consists into passing through depth sensors (*e.g.*, LiDARs), indeed optical flow fields can be obtained by projecting the 3D points from a given frame into the next frame [10], although it cannot take into account independently moving objects, for which manual post-processing or annotation remains necessary [35, 37]. The literature is rich of self-supervised strate-

*Joint first authorship.

gies [34, 30, 28, 23] from unlabeled videos to soften this constraint, but they mostly excel when deployed on data similar to those observed for training, a scenario unlikely to occur in most real applications.

Given both the aforementioned domain shift issue and the lack of real imagery annotated for optical flow, we propose an alternative scheme to distill proxy labels from real images for effective training of optical flow estimation networks. Following the observation that depth is required to obtain dense matching across views through reprojection [10, 35, 37], we use a monocular depth estimation network to revert the annotation process: given a single image and its estimated depth, we suppose a *virtual* motion of the camera to compute a dense optical flow field and, consequently, synthesize a new *virtual* image accordingly. For instance, in Figure 1 from a) pictures of a person and a cat, we estimate b) monocular depth and generate c) a flow field used to synthesize d) a novel view. We dub this process *Depthstillation*, and any single image is eligible for producing optical flow annotations through it.

Experiments carried out on synthetic (Sintel) and real (KITTI 2012 and 2015) datasets support our main claims:

- We show that it is possible to train an optical flow network on a collection of unrelated images, *e.g.* single pictures readily available online
- Using real images through our technique allows us to train networks that better transfer to real data than their counterparts trained on synthetic images, while finetuning these latter on *dephtstilled* frames and then on real data improves specialization
- Networks trained on our dephtstilled frames and flow labels better transfer to new real datasets than state-of-the-art self-supervised strategies using real videos [23]

2. Related Work

In this section, we review the literature relevant to the research topics touched by our work.

Optical Flow - Energy Minimization models. For a long time, optical flow has been cast as a continuous optimization problem through variational frameworks [16, 3, 67]. These approaches involve a data term coupled with regularization terms, and improvements to the former [4, 63] or the latter [44] represented the primary strategy to increase optical flow accuracy for years [50]. More recent strategies consider optical flow as a discrete optimization problem, despite managing the sizeable 2D search space required to determine corresponding pixels between images [36, 6, 65] is challenging. Until a few years ago [7], early attempts to improve optical flow with deep networks mainly focused on learning more robust data terms by training CNNs to match patches across images [63, 1, 65].

End-to-end Optical Flow. FlowNet [7] is the first end-to-end deep architecture proposed for optical flow. Concurrently, to satisfy the massive amount of training data required, synthetic datasets with dense optical flow ground-truth labels were made available [7, 33]. Eventually, other architectures [20, 51, 52, 18, 19, 17, 53] further improved accuracy on popular synthetic [5, 33] and real [35, 10] benchmarks, with RAFT [53] representing state-of-the-art.

For most existing networks, generalization remains a cause of concerns, in particular when moving from synthetic [7, 33] to real images [10, 35]. With our work, we show how to generate plausible training samples from real, unrelated images allowing for superior generalization.

Self-supervised Optical Flow. Being ground-truths hard to obtain for real data, self-supervised strategies allow to relax this requirement [21, 46, 34]. More recent advances introduced teaching-student frameworks [29], occlusion generation [30] and transformed data from augmentation [28]. Jonschkowski *et al*. [23] highlighted the key components to achieve state-of-the-art results in this setting.

Most of these approaches train on unlabeled videos (*e.g.*, from the KITTI 2015 multiview dataset [35]) from the same domain where the evaluation is carried out (*e.g.*, the KITTI 2015 optical flow benchmark). In contrast, in our work, we relax both constraints of having i) organized video collections and ii) taken in similar domains, achieving superior generalization compared to self-supervised networks.

Single Image Depth Estimation. In parallel to supervised approaches [64, 25, 9], many works focused on self-supervised strategies, aimed at replacing ground-truth labels with collections of images, either relying on stereo pairs [11, 58, 62] or monocular videos [69, 12, 13, 59]. To improve generalization, recent works [26, 45] exploited supervision from a large variety of images and auxiliary strategies such as Multi-View Stereo methods [48].

Shared by all these methods is the assumption of static scenes, required for reprojection across multiple views. In this paper, we show how a network trained according to such a strategy allows for generating, from still images, training data that well model motions, to train optical flow networks that are effective in presence of moving objects.

Novel View Synthesis. View synthesis aims at creating new images observed from arbitrary viewpoints starting from a given scene. It is gaining an ever increasing interest in computer vision [66, 38, 8, 60, 47], and it is a fundamental step to address many other tasks, such as video interpolation [22, 2, 40] or 3D effects [49, 70, 41].

Conversely, we focus on creating image pairs and corresponding ground-truth pixel displacements rather than visually pleasant videos. While some of the techniques mentioned above rely on pre-trained flow networks [40], our goal is to generate data to train these latter.

Data distillation through depth estimation. Strictly re-

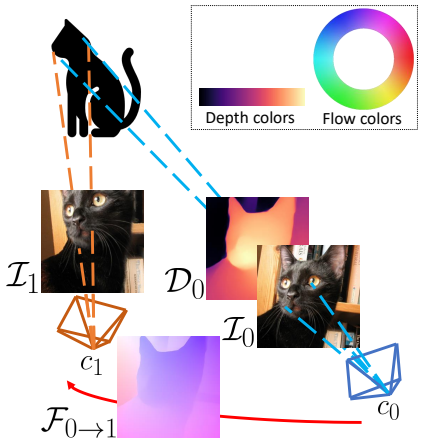


Figure 2. **Overview of the proposed depthstillation pipeline.** Given a single image $\mathcal{I}_0$ and its estimated depth map $\mathcal{D}_0$, we place the camera in c_0 and *virtually* move it (red arrow) towards a new viewpoint c_1. From the depth and virtual ego-motion, we obtain optical flow labels $\mathcal{F}_{0\to1}$ and a novel $\mathcal{I}_1$ through forward warping.

lated to our work is [61], estimating depth from single images to synthesize virtual right views and thus obtain stereo pairs, used to train deep stereo networks.

Despite the analogy of using single image depth estimation, we point out that our goal differs from [61] since we aim at modeling arbitrary motions in the scene (*i.e.*, optical flow) rather than a horizontal pixel displacement between synchronized images (*i.e.*, disparity). Purposely, we will describe the additional strategies required to attain, from single still images, the best training data for optical flow networks.

3. Depthstillation pipeline

In this section we illustrate our proposed framework to generate new virtual views $\mathcal{I}_1$ from single images $\mathcal{I}_0$, with corresponding dense optical flow ground-truth maps $\mathcal{F}_{0\to1}$. An overview of our pipeline is shown in Figure 2.

Virtual camera motion engine. Given $\mathcal{I}_0$, an off-the-shelf monocular depth network Φ is used to estimate its depth map $\mathcal{D}_0$

$$\mathcal{D}_0 = \Phi(\mathcal{I}_0) \tag{1}$$

used to project pixels in $\mathcal{I}_0$ to 3D space according to some plausible inverse intrinsics matrix K^{-1}. In case the network estimates inverse depth, we bring it to the depth domain first. $\mathcal{D}_0$ usually shows blurred edges [61, 49], causing flying pixels in the 3D space that can be easily sharpened via edge-preserving filters [32].

We now assume the camera used to frame image $\mathcal{I}_0$ to be at 3D location c_0 and apply an arbitrary *virtual* motion, moving it towards a new position c_1. To this aim, we generate a plausible rotation R_1 by sampling a random triplet of Euler angles and a plausible translation t_1 by sampling a random 3D vector. Then, we obtain the transformation matrix $T_{0\to1} = (R_1|t_1)$ corresponding to such roto-translation. Thus, we can project our 3D points to the image space through K in order to obtain a new image $\mathcal{I}_1$. This allows to obtain, for each pixel p_0 in $\mathcal{I}_0$, the coordinates p_1 of its corresponding pixel in $\mathcal{I}_1$ acquired from viewpoint c_1

$$p_1 \sim K T_{0\to1} \mathcal{D}_0(p_0) K^{-1} p_0 \tag{2}$$

and flow $\mathcal{F}_{0\to1}$ is obtained as the difference between p_1 and p_0. We point out that $\mathcal{F}_{0\to1}$ only models the virtual camera ego-motion, *i.e.* no object has moved independently. Finally, we obtain the new image $\mathcal{I}_1$ through forward warping.

Forward warping suffers from two well-known problems [61], that are *collisions* (*i.e.*, multiple pixels from $\mathcal{I}_0$ being warped to the same location in $\mathcal{I}_1$) and *holes* (*i.e.*, pixels in $\mathcal{I}_1$ over which no pixel from $\mathcal{I}_0$ is projected). To handle collisions, we keep track of pixels p_1 having multiple projections p_0 in a binary collision mask $\mathcal{M}$ (*i.e.*, collisions are labeled as 1, other pixels as 0) and select, for each, the one having minimum depth according to camera in position c_1, *i.e.* the closest, to be displayed in $\mathcal{I}_1$.

Hole filling. Artefacts introduced by holes are more subtle to be solved. Moreover, applying a 6DoF transformation to the camera plane vastly increases the chance of occurrence of holes compared to the case of 1D camera translations applied to distill stereo pairs [61]. In particular, in case of larger camera motion/rotations some *stretching* artefacts occur on the foreground objects (and, occasionally, in the background as well) as shown in Figure 3 a). To remove these holes, we build a binary hole mask $\mathcal{H}$, as in Figure 3 b), where we label pixels in $\mathcal{I}_1$ for which no pixel in $\mathcal{I}_0$ is reprojecting on to with 0. Then, a simple inpainting strategy [54] is usually sufficient to fill them, as reported in Figure 3 c) on the girl's face. Unfortunately, this is not enough in the case of stretching artefacts occurring in a foreground object overlapping a background one. Indeed, in this case, it is very likely that the holes induced by the stretching of the foreground object are filled by pixels in the background. These pixels are not detected by $\mathcal{H}$, causing the bleeding effect shown in Figure 3 c), where the hair merges with the background umbrella. Since most of these artefacts occur

a) b) c) d) e)

Figure 3. **Hole filling strategies.** From left to right: a) forward-warped image affected by stretching artefacts, b) holes mask $\mathcal{H}$ c) inpainted image, d) collision-augmented holes mask $\mathcal{H}'$ and e) improved inpainted image. Black pixels in $\mathcal{H}$ and $\mathcal{H}'$ are those to be inpainted.

in non-colliding pixels surrounded by colliding ones, *i.e.* in $\mathcal{M}$ they are labeled as 0 and surrounded by pixels labeled as 1, we can detect them by dilating $\mathcal{M}$ into $\mathcal{M}'$. Then, we define the binary mask $\mathcal{P}$ assigning 1 to pixels having the same label in $(\mathcal{M}', \mathcal{M})$ and 0 to the remaining (*i.e.* those that become 1 in $\mathcal{M}'$). We finally obtain $\mathcal{H}'$ by multiplying $\mathcal{H}$ and $\mathcal{P}$

$$\mathcal{P} = (\mathcal{M}' == \mathcal{M}), \qquad \mathcal{H}' = \mathcal{H} \cdot \mathcal{P} \tag{3}$$

We can apply the inpainting algorithm to pixels labeled with 0 in $\mathcal{H}'$, shown in Figure 3 d), to obtain Figure 3 e), where the foreground-background bleeding does not occur. We report more qualitative examples regarding the different masks in the supplementary material.

We point out how, in large dis-occluded area (*i.e.*, in the proximity of depth boundaries, as shown in Figure 3 on the left of the person), the inpainting method produces blurred content, as shown in Figure 3 c) and e). Despite these artefacts, our experiments will prove that hole filling improves the accuracy of trained networks significantly. We report in the supplementary material additional qualitative results concerning the design choices discussed so far.

Independent motions. The pipeline sketched so far models the optical flow field occurring between images acquired in a static environment, *i.e.* consequence of the camera motion, not taking into account possible independently moving objects, very likely to occur in real contexts [35]. In order to model more realistic simulations, we introduce the possibility of applying different virtual motions to objects extracted from the scene by leveraging an instance segmentation network Ω for extracting N objects Π_i, $i \in [1,\text{N}]$

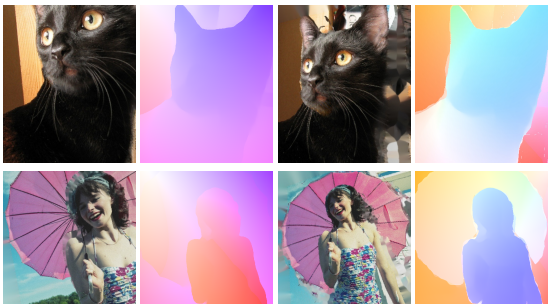
$$\Pi = \{\Pi_i, i \in [1, \text{N}]\} = \Omega(\mathcal{I}_0) \tag{4}$$

Then, to simulate a motion of the object in the scene, we randomly move the camera from c_0 towards a point $c_{\pi_i} \neq c_1$ and its corresponding transformation $T_{0 \to \pi_i}$ to be applied to object Π_i. Then, we reproject pixels from $\mathcal{I}_0$ on the image planes of the different cameras. Pixel coordinates in $\mathcal{I}_1$ will be selected according to their belonging to segmented objects or the background as

$$p_1 \sim \begin{cases} KT_{0\to1}\mathcal{D}_0(p_0)K^{-1}p_0 & \text{if } p_0 \notin \Pi \\ KT_{0\to\pi_i}\mathcal{D}_0(p_0)K^{-1}p_0 & \text{if } p_0 \in \Pi_i \end{cases} \tag{5}$$

We handle collisions as outlined before, keeping pixels whose depth results lower after motion. Finally, we obtain optical flow $\mathcal{F}_{0\to1}$ and image $\mathcal{I}_1$ as aforementioned.

To be robust to noisy/false detections, *e.g.* in case of tiny blobs accidentally labeled as objects, we rank the objects according to their size, *i.e.* number of pixels, and keep in Π only the n <N largest objects. Figure 4 shows two qualitative comparisons between images and flow distilled by merely applying a virtual camera motion, a) and b), and those obtained by segmenting the cat or the person in the foreground and simulating an independent motion, c) and d). Although our formulation simulates moving objects by moving virtual cameras instead, we can notice how the final effect on $\mathcal{I}_1$ and $\mathcal{F}_{0\to1}$ is equivalent for our purposes.



a) b) c) d)

Figure 4. **Independent motions modeling.** From left to right: a) image generated by only modeling camera motion and b) corresponding optical flow field, c) image generated after segmenting the foreground, which is now subject to a different motion yielding d) a more complex optical flow field.

We point out that, by increasing the number of moving objects, collisions and holes increase. In particular, a higher number of dis-occlusions might appear after applying independent motions, leading to blurry inpainted content, as shown in Figure 4 c) on the top row, on the right of the cat. Besides, shape boundaries may be inconsistent across depth and segmentation predictions, afflicting the truthfulness of the generated image and introducing artefacts (e.g., background pixels moved as part of the foreground). We will see how, although helpful, this approach yields minor improvements compared to the previous two steps performed in our framework, that result crucial for dephtstilling reliable training data. Moreover, segmenting object instances requires an additional network Ω trained in a supervised manner conversely to single-image depth estimation networks, whereby an extensive literature of self/weakly-supervised approaches exists [11, 12, 58, 62].

4. Experimental results

In this section, we describe the experimental setup used to validate our depthstillation pipeline. The source code is available at https://github.com/mattpoggi/depthstillation.

4.1. Training datasets

At first, we describe the datasets used to train the networks considered in our experiments.

Chairs (Ch). FlyingChairs [7] is a popular synthetic dataset used to train optical flow models. It contains 22232

images of chairs moving according to 2D displacement vectors over random backgrounds sampled from Flickr.

Things (Th). The FlyingThings3D dataset [20] is a collection of 3D synthetic scenes belonging to the SceneFlow dataset [33] and contains a training split made of 19635 images. Differently from Chairs, objects move in the scene with more complex 3D motions. State-of-the-art networks usually train in sequence over Chairs and Things (Ch→Th).

COCO dataset. The COCO dataset [27] is a collection of single still images (it provides $\mathcal{I}_0$ only) and ground-truth with labels for tasks such as object detection or panoptic segmentation, but lacks any depth or optical flow annotation. We sample images from the *train2017* split, which contains 118288 pictures, to generate virtual images and optical flow maps. We dub *dephtstilled* COCO (**dCOCO**) the training set obtained in such a manner.

DAVIS. The DAVIS dataset [42] provides high-resolution videos and it is widely used for video object segmentation. Since it does not provide optical flow ground-truth labels, we use all the 10581 images of the unsupervised 2019 challenge to generate **dDAVIS** and compare with the state-of-the-art in self-supervised optical flow [23].

4.2. Testing datasets

We describe here the testing imagery used to evaluate the networks trained on the datasets mentioned above. As metrics, we report the average End-Point Error (EPE) and two error rates, respectively the percentage of pixels with absolute error greater than 3 ($>$ 3) or both absolute and relative errors greater than 3 and 5% respectively (Fl), as defined in [35], on *All* pixels. In every experiment, we will highlight the best results in **bold** and underline the second-best among methods trained in fair conditions.

Sintel. Sintel [5] is a synthetic dataset with ground-truth optical flow maps. We use its training split, counting 1041 images for both Clean and Final passes, for evaluation.

KITTI. The KITTI dataset is a popular dataset for autonomous driving with sparse ground-truth values for both depth and optical flow tasks. Two versions exist, KITTI 2012 [10] counting 194 images framing static scenes and KITTI 2015 [35] made of 200 images framing moving objects, in both cases gathered by a car in motion.

4.3. Implementation Details

We describe next our pipeline and the networks used for depth estimation and learning optical flow.

Depth estimation models. To obtain dense depth maps from single RGB images, we select two models, respectively MiDaS [45] and MegaDepth [26], the former because represents the state-of-the-art for depth estimation in-the-wild and the latter because trained with weaker supervision than MiDaS[1]. Next, we will show how the accuracy of networks trained on our data is affected by the depth estimator.

Depthstillation pipeline. To generate virtual images, we convert predicted depths into $[1, 100]$. Given a single image of resolution W×H, we assume a virtual camera having fixed K, with focals $(f_x, f_y) = 0.58$(W,H) and optical center $(c_x, c_y) = 0.5$(W,H). To generate $T_{0\to1}$, we build t_1 by sampling three scalars t_x, t_y, t_z in $[-0.2, 0.2]$ and R_1 by sampling three Euler angles in $[-\frac{\pi}{18}, \frac{\pi}{18}]$. To simulate moving objects, we run pre-trained Mask-RCNN [14] to select $n = 2$ instance masks and generate t_i and R_i sampling respectively in $[-0.1, 0.1]$ and $[-\frac{\pi}{36}, \frac{\pi}{36}]$ and add them to R_1 and t_1. Depth maps are sharpened by means of 2 iterations of a 5×5 bilateral filter, while we dilate $\mathcal{M}$ with a 3×3 kernel. We can generate multiple camera motions for any given single image and thus a variety of pairs and ground-truth labels. We will see how playing with the number of images and motions impacts optical flow network accuracy.

Optical Flow networks. To evaluate how effective our distilled images are at training optical flow models, we select two main architectures: RAFT [53] and PWC-Net [51]. The first because it represents state-of-the-art architecture for supervised optical flow, already enabling excellent generalization capability. The second because it achieves the best results among self-supervised methods (*e.g.*, UFlow [23]). By deploying both architectures, we aim to prove that our method is general and significantly improves generalization in supervised and self-supervised optical flow. When not otherwise specified, we train RAFT on depthstilled data for 100K steps with a learning rate of 4×10^{-4} and weight decay of 10^{-4}, batch size of 6 and 496×368 image crops. This configuration is the largest one fitting into a single NVIDIA Titan X GPU. Following [53], we adopted AdamW as optimizer [31] and applied the same data augmentations and loss functions, while we set 12 as the number of iterative updates. To train PWCNet, we used as optimizer Adam [24], with an initial learning rate of $1e^{-4}$ and halved after 400K, 600K and 800K steps. We trained our model for 1M steps with a batch size of 8, adopting the multi-scale loss used in [51] for the synthetic pre-training, with the same augmentations and crop size used for RAFT.

4.4. Ablation Study

In this section, we assess the impact of the different components of our pipeline.

Depth, hole filling and moving objects. We start by ablating our pipeline to measure the impact of i) estimating depth, ii) applying hole filling to generated images and iii) simulating objects moving independently. This study is carried out by generating virtual views from 20K COCO im-

[1]The reader might argue that MiDaS has been trained on labels produced by pre-trained optical flow networks, introducing biases into images generated with our pipeline. However, we point out that optical flow networks are used only to handle negative disparities in stereo images and would not be necessary if, given the minimum negative disparity d_{min}, the right image is shifted left by $|d_{min}|$, thus making $d_{min} = 0$.

	Depth est.	Hole fill.	Moving obj.	Sintel C.		Sintel F.		KITTI 12		KITTI 15	
				EPE	>3	EPE	>3	EPE	Fl	EPE	Fl
(A)	✗	✗	✗	5.50	18.22	6.08	20.83	3.31	18.95	10.51	35.52
(B)	✓	✗	✗	2.52	7.17	3.72	11.04	2.02	7.53	4.84	16.26
(C)	✓	✓	✗	2.63	7.00	3.90	11.31	**1.82**	6.62	3.81	12.42
(D)	✓	✓	✓	**2.35**	**6.11**	**3.62**	**10.10**	1.83	**6.53**	**3.65**	**11.98**

Table 1. **Method ablation.** We train RAFT on dCOCO with different configurations of depthstillation: (A) constant depth for each image, (B) adding depth estimated by MiDaS [45], (C) adding hole-filling and (D) simulating object motions.

	Depth Model	Sintel C.		Sintel F.		KITTI12		KITTI15	
		EPE	>3	EPE	>3	EPE	Fl	EPE	Fl
(A)	No depth	5.50	18.22	6.08	20.83	3.31	18.95	10.51	35.52
(B)	Megadepth [26]	2.91	7.51	3.99	11.55	**1.81**	7.11	4.10	13.70
(C)	MiDaS [45]	**2.63**	**7.00**	**3.90**	**11.31**	1.82	**6.62**	**3.81**	**12.42**

Table 2. **Impact of depth estimator.** We train RAFT on dCOCO without depth estimation (A), using depth maps provided by MegaDepth (B) or MiDaS (C).

	# Training samples			Sintel C.		Sintel F.		KITTI12		KITTI15	
	Images	Motions	Total	EPE	>3	EPE	>3	EPE	Fl	EPE	Fl
(A)	4K	$\times 1$	4K	2.73	6.96	3.97	11.09	1.86	6.81	3.93	12.56
(B)	4K	$\times 5$	20K	2.56	6.78	3.88	10.99	**1.77**	**6.62**	3.93	12.57
(C)	20K	$\times 1$	20K	2.63	7.00	3.90	11.31	1.82	**6.62**	**3.81**	12.42
(D)	20K	$\times 5$	100K	**2.37**	**6.69**	**3.64**	**10.73**	1.79	6.79	3.82	**12.39**

Table 3. **Impact of images and virtual motions.** We train several RAFT models by changing the number of input images taken from COCO and the number of motions depthstilled for each one.

ages, applying a single virtual camera motion for each, by training RAFT [53] on them and evaluating the final model on Sintel, KITTI 2012 and KITTI 2015. Table 1 collects the outcome of this evaluation. On row (A), we show the performance achieved by generating images without estimating their depth, thus assuming a constant depth value for all pixels in any image. By moving to row (B), for which we use MiDaS [45] to estimate depth during the depthstillation process, we can notice considerable improvements in all metrics and datasets, with Fl score often more than halved. Nonetheless, generated images are affected by large holes and this does not allow for optimal performance. By enabling hole filling (C), the trained RAFT further improves its accuracy on real datasets. Finally, in (D), we show results by simulating objects moving independently, that further improves the results on Sintel. The benefit of this latter strategy is consistent on most metrics, although minor on real datasets such as KITTI 2012 and 2015 compared to the improvements obtained by (B) and (C), proving that the simple camera motion combined with depth is enough to obtain a robust optical flow network capable of generalizing to real environments. Moreover, as already pointed out, (D) also requires a trained instance segmentation network, which is hard to obtain for any possible dataset and would consequently constrain our pipeline. Thus, since our primary focus is on real environments, we choose (C) as the configuration for the following experiments.

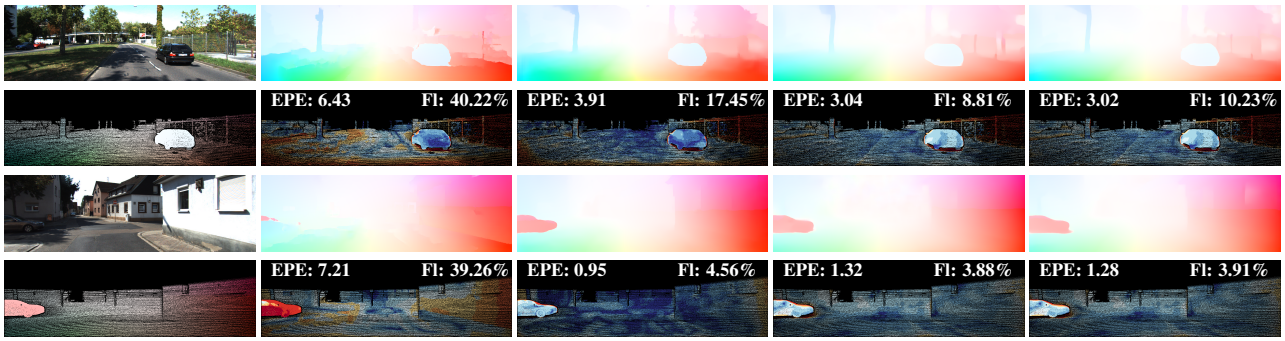
Depth estimation network. We measure the impact of the depth estimator on our overall data generation pipeline. To this aim, we follow the same protocol of the previous experiments, replacing MiDaS with MegaDepth [26] during the depth estimation step. Table 2 shows the results of this experiment. We can notice how images generated through MegaDepth (B) allow for training a RAFT model that places in between the one trained on images generated without depth (A) and using MiDaS (C), being much closer to the latter than to the former. This proves that depth is a crucial cue in our pipeline and the accuracy of the optical flow network, as we might expect, increases with the quality of the estimated depth maps, although with minor gains.

Amount of generated images. We can increase the amount of data we generate acting on two orthogonal dimensions: the number of images $\mathcal{I}_0$ and the number of virtual motions we simulate for each. Table 3 collects the results achieved by several RAFT models trained on a different number of images, obtained by varying the parameters mentioned above. By assuming 4K input images, we can notice how applying 5 virtual motions to each (B) allows a consistent boost on Sintel and KITTI 2012 compared to simulating a single motion each (A), while not improving on KITTI 2015. Interestingly, 4K images already allow for strong generalization to real domains, outperforming the results achieved using synthetic datasets shown in detail in the next section. On the other hand, increasing the input images by the same factor $\times 5$, yet simulating a single motion (C) leads worse results on Sintel while achieving some improvement on KITTI compared to (A) and (B). This fact highlights that a more variegate image content in the training dataset may be beneficial only for generalization to real environments. By depthstilling 5 motions, for a total of 100K training samples (D), yields further improvements on Sintel, again with minor impact on KITTI. To carry out a fair comparison with synthetic datasets, counting about 20K images each, we will use 20K images and a single virtual motion to depthstill our training data from now on.

4.5. Comparison with synthetic datasets

In this section, we evaluate the effectiveness of our depthstilled data versus synthetic datasets [7, 33].

Generalization to real environments. We start by evaluating the robustness of a network trained on our data when deployed on real datasets. Table 4 shows the performance achieved by RAFT when trained on Chairs (A) and fine-tuned on Things (B) with crop size and settings described in [53] to fit in a single GPU, compared to a variant trained on dCOCO, a split of 20K image pairs depthstilled from COCO (C). For completeness, we also report the performance of RAFT models provided by the authors (A†) and (B†), trained on $2\times$ GPUs and thus not directly comparable with our setting. We can notice how training on dCOCO (C) allows for much higher generalization on real datasets such



a) b) c) d) e)

Figure 5. **Qualitative results on the KITTI 2015 training set.** On two rows: a) reference frame (top) and ground-truth flow (bottom), optical flow maps (top) by RAFT trained on b) Ch, c) Ch→Th, d) dCOCO and e) Ch→Th→dCOCO and error maps (bottom).

	Dataset	Sintel C.		Sintel F.		KITTI 12		KITTI 15	
		EPE	>3	EPE	>3	EPE	Fl	EPE	Fl
(A†)	Ch	2.26	7.35	4.51	12.36	4.66	30.54	9.84	37.56
(B†)	Ch→Th	1.46	4.40	2.79	8.10	2.15	9.30	5.00	17.44
(A)	Ch	2.36	7.70	4.39	12.04	5.14	34.64	10.77	41.08
(B)	Ch→Th	**1.64**	**4.71**	**2.83**	**8.67**	2.40	10.49	5.62	18.71
(C)	dCOCO	2.63	7.00	3.90	11.31	1.82	**6.62**	3.81	**12.42**
(D)	Ch→Th→dCOCO	1.88	5.31	3.23	9.26	**1.78**	7.00	**3.42**	13.08

Table 4. **Comparison with synthetic datasets – generalization.** Generalization achieved by RAFT when trained on synthetic data (A),(B), on our dCOCO dataset (C) and a combination of both (D). † are obtained with publicly available weights by [53] (2× GPUs).

as KITTI 2012 and 2015, at the cost of worse performance on the Sintel synthetic dataset. This latter result is not surprising because the images in Things are generated through computer graphics as those in Sintel, while generating virtual images from a real dataset (COCO) leads to superior generalization on real datasets (KITTI 2012 and 2015), also outperforming (A†) and (B†) despite the single GPU.

We also train RAFT sequentially on Chairs, Things and dCOCO (D). This setting improves the EPE achieved by (C) on KITTI 2012 and 2015 and turns out much more effective on Sintel with both metrics. This fact suggests that a combination of synthetic images with perfect ground-truth and virtual images with depthstilled labels might be beneficial for generalization purposes. Figure 5 shows some qualitative optical flow predictions and corresponding error maps obtained from the RAFT variants considered in Table 4. We report additional examples in the supplementary material.

Fine-tuning on real data. We evaluate the effect of pre-training on synthetic images or our generated frames when fine-tuning on a few real data with accurate ground-truth. To this aim, we fine-tune RAFT variants on the first 160 images of the KITTI 2015 training set and evaluate on the remaining 40 and KITTI 2012. We train with a learning rate of 10^{-4} and weight decay of 10^{-5}, batch size of 3 and 960×288 image crops, converging after 20K iterations. Table 5 collects the outcome of this experiment. We can

	Pre-training	Fine-tuning	KITTI12		KITTI15	
			EPE	Fl	EPE	Fl
(A)	Ch	✗	5.14	34.64	15.56	47.29
	Ch	✓	1.42	4.86	2.40	8.49
(B)	Ch→Th	✗	2.40	10.49	9.04	25.53
	Ch→Th	✓	1.36	4.67	2.22	8.09
(C)	dCOCO	✗	1.82	6.62	5.09	16.72
	dCOCO	✓	1.37	4.70	2.76	9.15
(D)	Ch→Th→dCOCO	✗	1.78	7.00	4.82	18.03
	Ch→Th→dCOCO	✓	**1.32**	**4.54**	**2.21**	**7.93**

Table 5. **Comparison with synthetic datasets – fine-tuning.** Performance of RAFT variants pre-trained on synthetic datasets (A) and (B), on dCOCO (C) or both (D) when fine-tuned on a subset of 160 images from KITTI 2015, tested on KITTI 2012 and the remaining 40 images from KITTI 2015.

	Model	Dataset	Sintel C.		Sintel F.		KITTI12		KITTI15	
			EPE	>3	EPE	>3	EPE	Fl	EPE	Fl
(A)	PWCNet	Ch	3.33	-	4.59	-	5.14	28.67	13.20	41.79
(B)	PWCNet	Ch→Th	**2.55**	-	**3.93**	-	4.14	21.38	10.35	33.67
(C)	PWCNet	dCOCO	4.14	11.54	5.57	15.58	**3.16**	**13.30**	**8.49**	**26.06**
(D)	RAFT	dCOCO	2.63	7.00	3.90	11.31	1.82	6.62	3.81	12.42

Table 6. **Impact of depthstillation on different architectures**. Evaluation on PWCNet and RAFT. Entries with "-" are not provided in the original paper.

notice how variants (A) and (B) trained on synthetic data are greatly improved by the fine-tuning, while (C) achieves slightly lower accuracy after fine-tuning. Despite allowing for much higher generalization to real images, the supervision allowed by our method is *weaker* than the one obtained through real image pairs and perfect ground-truth. Thus, it is not surprising that networks trained from scratch to the end on perfect ground-truth might yield better accuracy. Nonetheless, combining synthetic data with our depthstilled images (D) allows for the best performance, confirming the findings from our previous experiments that a combination of the two worlds – synthetic data with perfect labels and realistic yet imperfect images and labels – is beneficial.

Impact on different optical flow networks. To prove that the superior generalization we achieve is enabled by our

data rather than a specific architecture such as RAFT, we also train PWCNet [51] on the 20K images generated from COCO. Table 6 shows how PWCNet trained on dCOCO (C) dramatically outperforms the original variants trained on Chairs (A) and fine-tuned on Things (B) when testing on real data, at the cost of lower performance on Sintel synthetic images, substantially confirming our findings from previous experiments with RAFT, reported in the table for comparison (D). This fact proves that our data, generated from single yet realistic still images, significantly improves generalization to real data independently from the optical flow model trained.

4.6. Comparison with self-supervision from videos

Given the rich literature about self-supervised optical flow [34, 29, 30, 23], we compare our strategy with state-of-the-art practises for self-supervised optical flow [23].

Generalization. In contrast to most works in this field that train and test in the same domain [34, 29, 30, 23], we inquire about how well networks trained in a self-supervised manner or leveraging our proposal transfer across different real datasets. To this aim, we adopt DAVIS [42] for training and evaluate on KITTI 2012 and 2015 as in the previous experiments. To train UFlow [23], we use the official code provided by the authors. In particular, we trained the model on the entire DAVIS dataset for 1M steps, using a batch size of 1 as suggested in [23], 512×384 resized images and letting unchanged other configuration parameters in order to replicate the authors' settings. Being UFlow based on PWCNet, we train from scratch another instance of PWCNet on dDAVIS for the same number of steps with a batch of 8 over depthstilled images and labels. The learning rate scheduling is the same highlighted in section 4.3, while the crop is 512×384. This way, we evaluate how well a PWCNet trained on depthstilled data transfers to other datasets compared to a model trained on real videos framing the same image content of the depthstilled images. Table 7 collects the outcome of this experiment. We can notice how the PWCNet model trained on dDAVIS (B) transfers much better to the KITTI 2012 and 2015 datasets compared to UFlow trained on the real DAVIS (A), thanks to the stronger supervision from the distilled optical flow labels. For the sake of completeness, we also report the results achieved by RAFT (C) trained on the same data, confirming to be superior.

Limitations. Our pipeline has some obvious limitations. Indeed, the training samples we generate are far from being utterly realistic because cannot model some behaviors, such as the large 3D rotation of objects in the scene, frequently found in real videos. Thus, despite the strong generalization we achieve compared to self-supervision, real videos allow for much better specialization when training and testing in the same domain. As shown in Table 8, UFlow trained on the 4K images of the KITTI multiview dataset (A) performs much better than PWCNet trained on 960×288 crops from dKITTI (B), a set of about 4K images depthstilled from KITTI 2015 multiview testing set. On the other hand, RAFT trained on dKITTI with the same crop size (C) gets closer to UFlow, thanks to the more effective architecture.

	Model	Dataset	KITTI12		KITTI15	
			EPE	Fl	EPE	Fl
(A)	UFlow	DAVIS	3.49	14.54	9.52	25.52
(B)	PWCNet	dDAVIS	**2.81**	**11.29**	**6.88**	**21.87**
(C)	RAFT	dDAVIS	1.78	6.85	3.80	13.22

Table 7. **Comparison between self-supervision and depthstillation – generalization.** Effectiveness of the two strategies when evaluated on unseen data (KITTI 2012 and 2015).

	Model	Dataset	KITTI12		KITTI15	
			EPE	Fl	EPE	Fl
(A)	UFlow	KITTI	-	-	**3.08**	**10.00**
(B)	PWCNet	dKITTI	2.64	9.43	7.92	22.17
(C)	RAFT	dKITTI	1.76	5.91	4.01	13.35

Table 8. **Comparison between self-supervision and depthstillation – specialization.** Effectiveness of the two strategies when training and testing on similar data (KITTI 2015). Entries with "-" are not provided in the original paper.

This lower specialization is also due to the completely random motions we depthstill. In contrast, KITTI motions consist of a much smaller subset (*i.e.* mostly forward translations or steerings) dominant in the real KITTI multiview split, yet rarely occurring in dKITTI.

As take-home message, our depthstillation strategy effectively addresses the scarcity of training data, *e.g.* when annotated images or not-annotated videos of the target environment are not available, yielding superior generalization compared to existing practices. Moreover, it is complementary to domain-specific real training data with labels, seldom ever available in practice.

5. Conclusion

We proposed a new strategy named, *Depthstillation*, to distill dense optical flow ground-truth maps from single still images and create novel virtual views, by leveraging the depth provided by a pre-trained monocular network. Through extensive experiments, we showed how it allows for training state-of-the-art optical flow networks [51, 53], leading to models that better generalize to real data compared to the use of synthetic images or self-supervision from videos framing different content, while suffering at specialization. Depthstillation is a powerful solution when domain-specific training data is not available, as occurs in most practical applications in-the-wild.

Acknowledgement. We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan Xp GPU used for this research.

References

[1] Min Bai, Wenjie Luo, Kaustav Kundu, and Raquel Urtasun. Exploiting semantic information and deep matching for optical flow. In *European Conference on Computer Vision*, pages 154–170. Springer, 2016. 1, 2

[2] Wenbo Bao, Wei-Sheng Lai, Chao Ma, Xiaoyun Zhang, Zhiyong Gao, and Ming-Hsuan Yang. Depth-aware video frame interpolation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3703–3712, 2019. 2

[3] Michael J Black and Padmanabhan Anandan. A framework for the robust estimation of optical flow. In *1993 (4th) International Conference on Computer Vision*, pages 231–236. IEEE, 1993. 2

[4] Thomas Brox, Christoph Bregler, and Jitendra Malik. Large displacement optical flow. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 41–48. IEEE, 2009. 2

[5] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black. A naturalistic open source movie for optical flow evaluation. In A. Fitzgibbon et al. (Eds.), editor, *European Conf. on Computer Vision (ECCV)*, Part IV, LNCS 7577, pages 611–625. Springer-Verlag, Oct. 2012. 1, 2, 5

[6] Qifeng Chen and Vladlen Koltun. Full flow: Optical flow estimation by global optimization over regular grids. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4706–4714, 2016. 2

[7] Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philip Hausser, Caner Hazirbas, Vladimir Golkov, Patrick Van Der Smagt, Daniel Cremers, and Thomas Brox. Flownet: Learning optical flow with convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2758–2766, 2015. 1, 2, 4, 6

[8] John Flynn, Michael Broxton, Paul Debevec, Matthew DuVall, Graham Fyffe, Ryan Overbeck, Noah Snavely, and Richard Tucker. Deepview: View synthesis with learned gradient descent. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2367–2376, 2019. 2

[9] Huan Fu, Mingming Gong, Chaohui Wang, Kayhan Batmanghelich, and Dacheng Tao. Deep ordinal regression network for monocular depth estimation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2

[10] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012. 1, 2, 5

[11] Clément Godard, Oisin Mac Aodha, and Gabriel J Brostow. Unsupervised monocular depth estimation with left-right consistency. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 270–279, 2017. 2, 4

[12] Clément Godard, Oisin Mac Aodha, Michael Firman, and Gabriel J. Brostow. Digging into self-supervised monocular depth prediction. In *The International Conference on Computer Vision (ICCV)*, October 2019. 2, 4

[13] Vitor Guizilini, Rares Ambrus, Sudeep Pillai, Allan Raventos, and Adrien Gaidon. 3d packing for self-supervised monocular depth estimation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2

[14] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017. 5

[15] Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei Efros, and Trevor Darrell. Cycada: Cycle-consistent adversarial domain adaptation. In *International conference on machine learning*, pages 1989–1998. PMLR, 2018. 1

[16] Berthold KP Horn and Brian G Schunck. Determining optical flow. In *Techniques and Applications of Image Understanding*, volume 281, pages 319–331. International Society for Optics and Photonics, 1981. 2

[17] Tak-Wai Hui and Chen Change Loy. LiteFlowNet3: Resolving Correspondence Ambiguity for More Accurate Optical Flow Estimation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020. 1, 2

[18] Tak-Wai Hui, Xiaoou Tang, and Chen Change Loy. LiteFlowNet: A Lightweight Convolutional Neural Network for Optical Flow Estimation. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8981–8989, 2018. 1, 2

[19] Tak-Wai Hui, Xiaoou Tang, and Chen Change Loy. A lightweight optical flow cnn - revisiting data fidelity and regularization. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020. 1, 2

[20] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox. Flownet 2.0: Evolution of optical flow estimation with deep networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jul 2017. 1, 2, 5

[21] J Yu Jason, Adam W Harley, and Konstantinos G Derpanis. Back to basics: Unsupervised learning of optical flow via brightness constancy and motion smoothness. In *ECCV Workshops (3)*, 2016. 2

[22] Huaizu Jiang, Deqing Sun, Varun Jampani, Ming-Hsuan Yang, Erik Learned-Miller, and Jan Kautz. Super slomo: High quality estimation of multiple intermediate frames for video interpolation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9000–9008, 2018. 2

[23] Rico Jonschkowski, Austin Stone, Jon Barron, Ariel Gordon, Kurt Konolige, and Anelia Angelova. What matters in unsupervised optical flow. *ECCV*, 2020. 2, 5, 8

[24] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 5

[25] Iro Laina, Christian Rupprecht, Vasileios Belagiannis, Federico Tombari, and Nassir Navab. Deeper depth prediction with fully convolutional residual networks. In *3DV*, 2016. 2

[26] Zhengqi Li and Noah Snavely. Megadepth: Learning single-view depth prediction from internet photos. In *Computer Vision and Pattern Recognition (CVPR)*, 2018. 2, 5, 6

[27] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence

Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014. 5

[28] Liang Liu, Jiangning Zhang, Ruifei He, Yong Liu, Yabiao Wang, Ying Tai, Donghao Luo, Chengjie Wang, Jilin Li, and Feiyue Huang. Learning by analogy: Reliable supervision from transformations for unsupervised optical flow estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6489–6498, 2020. 2

[29] Pengpeng Liu, Irwin King, Michael R Lyu, and Jia Xu. Ddflow: Learning optical flow with unlabeled data distillation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 8770–8777, 2019. 2, 8

[30] Pengpeng Liu, Michael Lyu, Irwin King, and Jia Xu. Selflow: Self-supervised learning of optical flow. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4571–4580, 2019. 2, 8

[31] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. 5

[32] Ziyang Ma, Kaiming He, Yichen Wei, Jian Sun, and Enhua Wu. Constant time weighted median filtering for stereo matching and beyond. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 49–56, 2013. 3

[33] Nikolaus Mayer, Eddy Ilg, Philip Hausser, Philipp Fischer, Daniel Cremers, Alexey Dosovitskiy, and Thomas Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4040–4048, 2016. 2, 5, 6

[34] Simon Meister, Junhwa Hur, and Stefan Roth. UnFlow: Unsupervised learning of optical flow with a bidirectional census loss. In *AAAI*, 2018. 2, 8

[35] Moritz Menze and Andreas Geiger. Object scene flow for autonomous vehicles. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 1, 2, 4, 5

[36] Moritz Menze, Christian Heipke, and Andreas Geiger. Discrete optimization for optical flow. In *German Conference on Pattern Recognition*, pages 16–28. Springer, 2015. 2

[37] Moritz Menze, Christian Heipke, and Andreas Geiger. Object scene flow. *ISPRS Journal of Photogrammetry and Remote Sensing (JPRS)*, 2018. 1, 2

[38] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020. 2

[39] Zak Murez, Soheil Kolouri, David Kriegman, Ravi Ramamoorthi, and Kyungnam Kim. Image to image translation for domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4500–4509, 2018. 1

[40] Simon Niklaus and Feng Liu. Softmax splatting for video frame interpolation. In *IEEE International Conference on Computer Vision*, 2020. 2

[41] Simon Niklaus, Long Mai, Jimei Yang, and Feng Liu. 3d ken burns effect from a single image. *ACM Transactions on Graphics (TOG)*, 38(6):1–15, 2019. 2

[42] F. Perazzi, J. Pont-Tuset, B. McWilliams, L. Van Gool, M. Gross, and A. Sorkine-Hornung. A benchmark dataset and evaluation methodology for video object segmentation. In *Computer Vision and Pattern Recognition*, 2016. 5, 8

[43] Pierluigi Zama Ramirez, Alessio Tonioni, Samuele Salti, and Luigi Di Stefano. Learning across tasks and domains. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 8110–8119, 2019. 1

[44] René Ranftl, Kristian Bredies, and Thomas Pock. Non-local total generalized variation for optical flow estimation. In *European Conference on Computer Vision*, pages 439–454. Springer, 2014. 2

[45] René Ranftl, Katrin Lasinger, David Hafner, Konrad Schindler, and Vladlen Koltun. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2020. 2, 5, 6

[46] Zhe Ren, Junchi Yan, Bingbing Ni, Bin Liu, Xiaokang Yang, and Hongyuan Zha. Unsupervised deep learning for optical flow estimation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31, 2017. 2

[47] Gernot Riegler and Vladlen Koltun. Free view synthesis. In *European Conference on Computer Vision*, 2020. 2

[48] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 2

[49] Meng-Li Shih, Shih-Yang Su, Johannes Kopf, and Jia-Bin Huang. 3d photography using context-aware layered depth inpainting. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2, 3

[50] Deqing Sun, Stefan Roth, and Michael J Black. Secrets of optical flow estimation and their principles. In *2010 IEEE computer society conference on computer vision and pattern recognition*, pages 2432–2439. IEEE, 2010. 1, 2

[51] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8934–8943, 2018. 1, 2, 5, 8

[52] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. Models matter, so does training: An empirical study of cnns for optical flow estimation. *IEEE transactions on pattern analysis and machine intelligence*, 42(6):1408–1423, 2019. 1, 2

[53] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *European Conference on Computer Vision (ECCV)*, 2020. 1, 2, 5, 6, 7, 8

[54] Alexandru Telea. An image inpainting technique based on the fast marching method. *Journal of graphics tools*, 9(1):23–34, 2004. 3

[55] Marco Toldo, Andrea Maracani, Umberto Michieli, and Pietro Zanuttigh. Unsupervised domain adaptation in semantic segmentation: a review. *arXiv preprint arXiv:2005.10876*, 2020. 1

[56] Alessio Tonioni, Matteo Poggi, Stefano Mattoccia, and Luigi Di Stefano. Unsupervised adaptation for deep stereo. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1605–1613, 2017. 1

[57] Alessio Tonioni, Fabio Tosi, Matteo Poggi, Stefano Mattoccia, and Luigi Di Stefano. Real-time self-adaptive deep stereo. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 1

[58] Fabio Tosi, Filippo Aleotti, Matteo Poggi, and Stefano Mattoccia. Learning monocular depth estimation infusing traditional stereo knowledge. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9799–9809, 2019. 2, 4

[59] Fabio Tosi, Filippo Aleotti, Pierluigi Zama Ramirez, Matteo Poggi, Samuele Salti, Luigi Di Stefano, and Stefano Mattoccia. Distilled semantics for comprehensive scene understanding from videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2020. 2

[60] Richard Tucker and Noah Snavely. Single-view view synthesis with multiplane images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 2

[61] Jamie Watson, Oisin Mac Aodha, Daniyar Turmukhambetov, Gabriel J. Brostow, and Michael Firman. Learning stereo from single images. In *European Conference on Computer Vision (ECCV)*, 2020. 1, 3

[62] Jamie Watson, Michael Firman, Gabriel J Brostow, and Daniyar Turmukhambetov. Self-supervised monocular depth hints. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2162–2171, 2019. 2, 4

[63] Philippe Weinzaepfel, Jerome Revaud, Zaid Harchaoui, and Cordelia Schmid. Deepflow: Large displacement optical flow with deep matching. In *Proceedings of the IEEE international conference on computer vision*, pages 1385–1392, 2013. 2

[64] Dan Xu, Wei Wang, Hao Tang, Hong Liu, Nicu Sebe, and Elisa Ricci. Structured attention guided convolutional neural fields for monocular depth estimation. In *CVPR*, 2018. 2

[65] Jia Xu, René Ranftl, and Vladlen Koltun. Accurate optical flow via direct cost volume processing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1289–1297, 2017. 1, 2

[66] Jae Shin Yoon, Kihwan Kim, Orazio Gallo, Hyun Soo Park, and Jan Kautz. Novel view synthesis of dynamic scenes with globally coherent depths from a monocular camera. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5336–5345, 2020. 2

[67] Christopher Zach, Thomas Pock, and Horst Bischof. A duality based approach for realtime tv-l 1 optical flow. In *Joint pattern recognition symposium*, pages 214–223. Springer, 2007. 2

[68] Feihu Zhang, Xiaojuan Qi, Ruigang Yang, Victor Prisacariu, Benjamin Wah, and Philip Torr. Domain-invariant stereo matching networks. *arXiv preprint arXiv:1911.13287*, 2019. 1

[69] Tinghui Zhou, Matthew Brown, Noah Snavely, and David G Lowe. Unsupervised learning of depth and ego-motion from video. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1851–1858, 2017. 2

[70] Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: Learning view synthesis using multiplane images. In *SIGGRAPH*, 2018. 2